zoNNscan: a boundary-entropy index for zone inspection of neural models

Adel Jaouen, Erwan Le Merrer

▶ **To cite this version:**

Adel Jaouen, Erwan Le Merrer. zoNNscan: a boundary-entropy index for zone inspection of neural models. MCS 2020 - Monte Carlo Search workshop, Jan 2021, Virtual, Japan. pp.1-8. hal-03118264

HAL Id: hal-03118264
https://hal.science/hal-03118264

Submitted on 22 Jan 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

zoNNscan: a boundary-entropy index for zone inspection of neural models

Adel Jaouen[1] and Erwan Le Merrer[2]

[1] Technicolor, France
[2] Univ Rennes, Inria, CNRS, Irisa, France

Abstract. The training of deep neural network classifiers results in decision boundaries whose geometry is still not well understood. This is in direct relation with classification problems such as so called *corner case* inputs. We introduce `zoNNscan`, an index that is intended to inform on the boundary uncertainty (in terms of the presence of other classes) around one given input datapoint. It is based on confidence entropy, and is implemented through Monte Carlo sampling in the multidimensional ball surrounding that input. We detail the `zoNNscan` index, give an algorithm for approximating it, and finally illustrate its benefits on three applications.

Keywords: Decision boundaries, classification uncertainty, entropy, application of Monte Carlo sampling.

1 Introduction

Measures provide crucial insights in all computer science domains, as far as they allow to quantify and then analyze specific problems [18], or to serve as a building block for designing better algorithms [8]. In the blooming field of machine learning leveraging deep neural networks, open problems remain that are related to the position of *decision boundaries* in classification tasks. Little is known about the geometrical properties of those [6, 16, 4]. Yet, the close proximity of some inputs to those boundaries is getting more problematic with the increasing amount of critical applications that are using deep neural networks. The case of self-driving cars is an illustration of a critical application, where corner-cases have been recently found in production models (leading to wrong decisions) [17]; the inputs that cause the erroneous classifications are depicted to be close to decision boundaries. There are also recent works that are voluntarily tweaking the decision boundaries around given inputs, so that ownership information can be embedded into a target model [11].

While measures are inherently in use with neural networks to evaluate the quality of the learned model over a given dataset, we find that there is a lack of an index that provides information on the neighborhood of given inputs, with regards to the boundaries of other classes. Visual inspections of suspicious zones, by plotting in 2D a given decision boundary and the considered inputs is possible

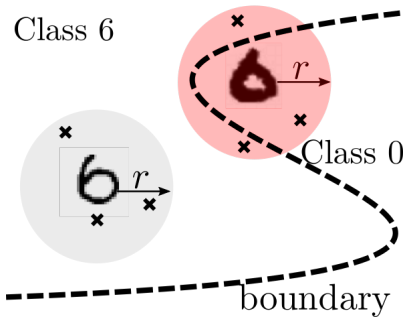

Fig. 1. An illustration of `zoNNscan` runs around two inputs. The example on the left (a 6 digit) is far from the model decision boundary, thus a `zoNNscan` at radius r returns a value close to 0. The example on the right is another 6 from the MNIST dataset, that is often misclassified (here as a 0, such as in [3]); its is close to the decision boundary of the 6 class. `zoNNscan` returns a higher value (up to 1) for it, as some data samples in the ball fall into the correct digit class.

[2]. In this paper, we propose an algorithm to output an index, for it helps to quantify possible problems at inference time.

2 Zoning classification uncertainty with `zoNNscan`

Classically, given an input and a trained model, the class predicted by the classifier is set to be the one with the higher score in the output vector [4]. These scores can thus be considered as membership probabilities to the model classes: output values close to 1 are to be interpreted as a high confidence about the prediction, and conversely, values close to 0 show high probabilities of non membership to the corresponding class. These scores also provide the information about the *uncertainty* of the model for a given input : vector scores close to each other indicate an uncertainty between these classes, and equal scores characterize a decision boundary point [13]. Thereby, a maximum uncertainty output refers to an input that causes inference to return an uniform vector of probabilities $\frac{1}{C}$ for a classification task into C-classes. Conversely, minimum uncertainty corresponds to an input that causes inference to return a vector of zeros, except for the predicted class for which a one is set. `zoNNscan` evaluates the uncertainty of predictions in a given input region. This is illustrated on Figure 1.

2.1 `zoNNscan`: definition

Given a discrete probability distribution $p = \{p_1, ..., p_C\} \in [0,1]^C$ with $\Sigma_{i=1}^{C} p_i = 1$ (C designates the number of events), we remind the definition of the Shannon entropy (b designates the base for the logarithm) :

$$\begin{aligned} H_b : [0,1]^C &\to \mathbb{R}^+ \\ p &\mapsto \Sigma_{i=1}^{C}(-p_i \log_b(p_i)). \end{aligned}$$

The maximum of H_b is reached in $p = \{\frac{1}{C}, ..., \frac{1}{C}\}$ and is equal to $\log_b(C)$ and with the convention $0 \log_b(0) = 0$, the minimum of H_b is 0. Subsequently, we use a C-based logarithm to make H_C output values in range $[0, 1]$. A classifier $\mathcal{M}$ is considered as a function defined on the input space $[0, 1]^d$ (on the premise that data are generally normalized for the training phase in such a way that all features are included in the range $[0, 1]$), taking its values in the space $\mathbb{P} = \{p \in [0, 1]^C \ s.t. \ \Sigma_{i=1}^{C} p_i = 1\}$. We introduce the composite function $\varphi = H_b \circ \mathcal{M}$ to model the indecision the network has on the input space. More specifically, we propose the expectation of $\varphi(U)$, $\mathbb{E}_{\mathbb{Z}}[\varphi(U)]$ (with U a uniform random variable on $\mathbb{Z}$), on an input zone $\mathbb{Z} \in [0, 1]^d$, to be an indicator on the uncertainty of the classifier on zone $\mathbb{Z}$.

Definition 1. *Let the* ***zoNNscan*** *index be, in zone* $\mathbb{Z}$*:*

$$\begin{aligned} &\mathcal{B}([0,1]^d) \to [0,1] \\ &\mathbb{Z} \qquad\quad \mapsto \mathbb{E}_{\mathbb{Z}}[\varphi(U)] = \int_{\mathbb{Z}} \varphi(u) f_U(u) \, \mathrm{d}u, \end{aligned}$$

where $\mathcal{B}(\mathbb{R}^d)$ *refers to the* $\mathbb{R}^d$ *Borel set and* f_U *the uniform density on* $\mathbb{Z}$*.*

The minima of $\mathbb{E}_{\mathbb{Z}}[\varphi(U)]$ depicts observations in $\mathbb{Z}$ that were all returning one confidence of 1, and $C-1$ confidences of 0. Conversely, the maxima indicates full uncertainty, where each observation returned $\frac{1}{C}$ confidence values in the output vector.

2.2 A Monte Carlo approximation of zoNNscan

In practice, as data are typically nowadays of high dimensionality (e.g., in the order of millions of dimensions for image applications) and deep neural networks are computing complex non-linear functions, one cannot expect to compute the exact value of this expectation. We propose a Monte Carlo method to estimate this expectation on a space corresponding to the surrounding zone of a certain input.

For inspection around a given input $X \in [0, 1]^d$, we consider a ball $\mathbf{B}$ of center X and radius r, as zone $\mathbb{Z}$ introduced in previous subsection. We perform a Monte Carlo sampling of k inputs in a ball for the infinite-norm, corresponding to the hyper-cube of dimension d, around X (as depicted on Figure 1). We are generating inputs applying random deformations[3] ϵ_i on each components X_i such as $max(-X_i, -r) \leq \epsilon_i \leq min(1 - X_i, r)$.

For instance, given a normalized input X and a positive radius r, Monte Carlo sampling is performed uniformly in the subspace of $\mathbb{R}^d$ defined as:

$$\mathbb{Z} = \mathbf{B_\infty}(X, r) \cap [0, 1]^d.$$

The larger the number of samples k, the better the approximation of the index; this value has to be set considering the application and the acceptable computation/time constraints for inspection.

[3] For more advanced sampling techniques in high dimensional spaces, please refer to e.g., [5].

Algorithm 1 Monte Carlo approximation of `zoNNscan`

Require: $\mathcal{M}, X, r, k$
  for $i = 0..k$ **do**
    $x' \leftarrow$ Monte Carlo Sampling in $\mathbf{B_\infty}(X, r) \cap [0, 1]^d$
    $P_i \leftarrow \mathcal{M}(x')$ {Inference}
    $entropy_i \leftarrow H_C(P_i)$
  end for
  return $\frac{1}{k} \sum_{i=1}^{k} entropy_i$

The `zoNNscan` index, for inspection of the surrounding of input X in a model $\mathcal{M}$, is presented on Algorithm 1.

3 Illustration use of `zoNNscan`

We now look at three example use cases for `zoNNscan`.

3.1 Uncertainty around a given input

A first `zoNNscan` use case is to assess at which distance and in which proportions are other classes surrounding an interesting input X.

We experiment using the multi-layer perceptron (noted MLP hereafter) proposed as an example in the Keras library [1], and with $k = 1,000$ for the remaining of the experiments. Figure 2 presents the `zoNNscan` values with respect various radii r, around two input examples (a 6 digit from MNIST and the image of a ship from the CIFAR-10 dataset). A radius value $r = 0$ is equivalent to the simple output vector corresponding to the inference of X, i.e., without actual sampling in $\mathbf{B}$. Our algorithm starts with a $r = 0$, up to 1[4]. For the top-experiment on the MNIST digit, the confidence value for $r = 0$ is high for digit 6 (around 0.9), which is reflected by `zoNNscan` value to be low (around 0.2). When r increases, the confidence values of other classes are increasing progressively, which results for `zoNNscan` increasing sharply; at radii of $r = [0.25, 1]$ uncertainty in such a ball is critical.

For the bottom-experiment on the ship image the CIFAR-10 dataset, we observe at $r = 0$ and close a higher value (around 0.25), that captures the relatively high score for the class 'cat'. The score remains around 0.25 up to $r = 0.30$, because other boundaries ('frog' and 'truck') are nearby. Finally `zoNNscan` tends to zero because the 'truck' class occupies the input space in its vast majority. This last fact illustrates that `zoNNscan` is based on probability vectors, and thus follows the dominant class in them. For the use of `zoNNscan` at inference time, it is of course more desirable to have low r values, so that the inspected zone remains confined around the input to examine.

[4] Note that for $X \in [0, 1]^d$ and for $r \geq 1$, $\mathbf{B_\infty}(X, r) \cap [0, 1]^d = [0, 1]^d$, then the space of normalized data is totally covered by the sampling process.

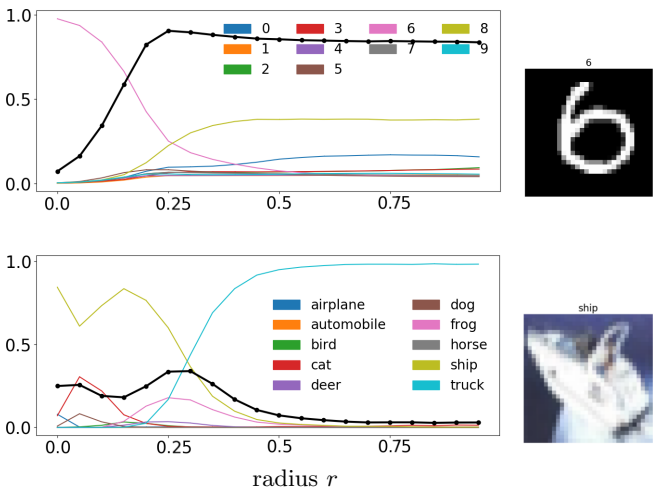

Fig. 2. `zoNNscan` values (dotted black line) on y-axes, with a varying radius $r \in [0, 1]$ on x-axes. Mean confidence scores for each class in the explored zone are also reported (colored lines). Top: MNIST dataset; a 6 digit classified by the MLP model. Bottom: CIFAR-10 dataset; a ship classified by RESNETv2.

This experiment brings two observations: *(*i) the study of the evolution of `zoNNscan` values with growing radii is of interest to characterize surrounding or problematic zones (e.g., sharp increase of the 'truck' class starting at 0.25 on Figure 2). *(*ii) for a r covering the whole input space, we obtain an estimate of the surface occupied by each class in the model (e.g., class 8 dominates on MNIST, while class 'truck' is covering most of classification space for the CIFAR-10 case).

3.2 Disagreement on corner case inputs

Arguably interesting inputs are the ones that make several deep neural network models disagree, as depicted in the case of self-driving cars [17]. Indeed, they constitute corner cases, that have to be dealt with for most critical applications. While this problem may be masked using voting techniques by several independent classifiers [9], we are interested in studying the `zoNNscan` value of those particular inputs.

We add two models for the experiment, also from Keras [1]: a convolutional neural network (CNN), and a recurrent neural network (IRNN), and focus on the MNIST dataset. Given two models, those inputs are extracted by comparing the predictions of the two models on a given dataset. In the MNIST dataset, we identify a total of 182 disagreements (i.e., corner cases) for the three models.

We plot on Figure 3 the two following empirical distributions. First, in blue, the distribution of $1,000$ `zoNNscan` values around test set inputs; second, in red, the distribution of the 182 corner case inputs. The `zoNNscan` index is also

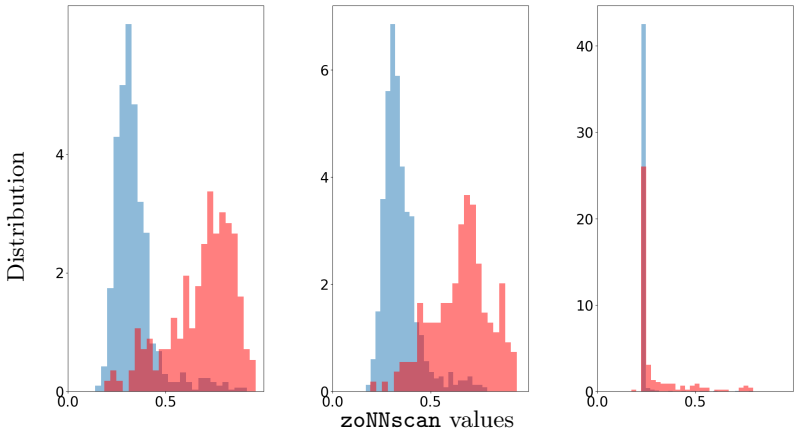

Fig. 3. Relative distributions of `zoNNscan` around 1,000 random test set inputs (blue) and the 182 corner case inputs (red). From left to right: the MLP, CNN and IRNN models.

estimated with $r = 0.025$. We observe a significant difference in the distributions (means are reported on Table 1).

In order to formally assess this difference in the two distributions, we performed the statistical *Kolmogorov-Smirnov 2-sample test* [15]. It tests the null hypothesis that two samples were drawn from the same continuous distribution. The p-values of this test correspond to the probabilities of observing the values under the null hypothesis. In any of the three case (MLP, CNN and IRNN models), the test rejects the null hypothesis with p-values lower than 10^{-3}.

3.3 Effects of watermarking on model boundaries

As a final illustration, we consider the recent work [11] consisting in *watermarking* a model. This operation is performed as follows: first, a watermark key consisting of inputs is created. Half of the key inputs are actually *adversarial examples* [7]. Second, the watermarking step consists in finetuning the model by retraining it over the key inputs, in order to re-integrate the adversarial examples in their original class. That last step is by definition moving the model decision boundaries, which constitutes the watermark that the model owner has sought to introduce.

All three models are watermarked with keys of size 100. Each input of those three keys is used as input X for `zoNNscan`, for 100 runs of Algorithm 1 with $k = 1,000$ samples each, $r = 0.05$. Two distributions of `zoNNscan` values are computed: one for the un-marked model, and another for the watermarked one.

If the watermarked model were indistinguishable from the original one, the two distributions would be the same. Once again, the Kolmogorov-Smirnov 2-sample test returns p-values lower than 10^{-3}. This reflects the fact that boundaries have changed around the key inputs, at least enough to be observed by

zoNNscan. As the watermarking process is expected to be as stealthy possible, this highlights that the key is to remain a secret for the model owner only [11].

Table 1. Mean of the empirical zoNNscan distributions, for the three neural networks.

Scenario	MLP	CNN	IRNN
1,000 inputs from test set	0.3709	0.3457	0.2403
182 corner case inputs	0.6857	0.6564	0.2989

4 Related works

While some metrics such as distance-from-boundaries [8] have been developed for support vector machines for instance, the difficulty to reason about the classifier decision boundaries of neural networks have motivated research works since decades. Lee et al. [12] proposed the extraction of features from decision boundaries, followed by [13] that targets the same operation without assuming underlying probability distribution functions of the data. An interesting library for visual inspection of boundaries, from low dimensional embedding is available online [2]. Fawzi et al. propose to analyze some geometric properties of deep neural network classifiers [6], including the curvature of those boundaries. Van den Berg [4] studies the decision region formation in the specific case of feedforward neural networks with sigmoidal nonlinearities.

If one can afford to use an *ensemble* of models, as well as joint *adversarial training* to assess the uncertainty of a model's decisions, the work in [10] provides an interesting approach; this makes it a good candidate for offline analysis of decisions. Cross entropy is one of the most basic measure for the training of neural networks. While it evaluates the divergence of the dataset distribution from the model distribution, it is not applied to inspect the confidence in the inference of specific zones in the input space. This paper addresses the relation of the surrounding of a given input with respect to the presence of decision boundaries of other classes; this relation is captured by the zoNNscan index, possibly at runtime, by leveraging the Shannon entropy.

5 Conclusion

We introduced a novel index, zoNNscan, for inspecting the surrounding of given inputs with regards to the boundary entropy measured in the zones of interest. We have presented a Monte Carlo estimation of that index, and have shown its applicability on a base case, on a technique for model watermarking, and on a concern regarding the adoption of current deep neural networks (corner case inputs). As this index is intended to be generic, we expect other applications to leverage it; future works include the study of the link of zoNNscan values with regards to other identified issues such as *trojaning* attacks on neural networks [14], and on the intensification of the Monte Carlo search.

References

1. Keras: Deep learning for humans. https://github.com/keras-team/keras/tree/master/examples, accessed: 2020-02-25
2. Plotting high-dimensional decision boundaries. https://github.com/tmadl/highdimensional-decision-boundary-plot, accessed: 2018-07-01
3. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (05 2002). https://doi.org/10.1109/34.993558
4. van den Berg, E.: Some insights into the geometry and training of neural networks. CoRR **abs/1605.00329** (2016)
5. Dick, J., Kuo, F.Y., Sloan, I.H.: High-dimensional integration: The quasi-monte carlo way. Acta Numerica **22**, 133–288 (2013). https://doi.org/10.1017/S0962492913000044
6. Fawzi, A., Moosavi-Dezfooli, S., Frossard, P., Soatto, S.: Classification regions of deep neural networks. CoRR **abs/1705.09552** (2017)
7. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. In: ICLR (2015)
8. Guo, G., Zhang, H.J., Li, S.Z.: Distance-from-boundary as a metric for texture image retrieval. In: IEEE International Conference on Acoustics, Speech, and Signal Processing. vol. 3, pp. 1629–1632 vol.3 (2001). https://doi.org/10.1109/ICASSP.2001.941248
9. Kuncheva, L., Whitaker, C., Shipp, C., Duin, R.: Limits on the majority vote accuracy in classifier fusion. Pattern Analysis & Applications **6**(1), 22–31 (Apr 2003)
10. Lakshminarayanan, B., Pritzel, A., Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems 30, pp. 6402–6413. Curran Associates, Inc. (2017), http://papers.nips.cc/paper/7219-simple-and-scalable-predictive-uncertainty-estimation-using-deep-ensembles.pdf
11. Le Merrer, E., Perez, P., Trédan, G.: Adversarial frontier stitching for remote neural network watermarking. Neural Comput & Applic 32, 92339244 (2020)
12. Lee, C., Landgrebe, D.A.: Feature extraction based on decision boundaries. IEEE Transactions on Pattern Analysis and Machine Intelligence **15**(4), 388–400 (Apr 1993)
13. Lee, C., Landgrebe, D.A.: Decision boundary feature extraction for neural networks. IEEE Transactions on Neural Networks **8**(1), 75–83 (Jan 1997). https://doi.org/10.1109/72.554193
14. Liu, Y., Ma, S., Aafer, Y., Lee, W.C., Zhai, J., Wang, W., Zhang, X.: Trojaning attack on neural networks. In: NDSS (2018)
15. Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Statist. **18**(1), 50–60 (03 1947). https://doi.org/10.1214/aoms/1177730491
16. Moosavi-Dezfooli, S., Fawzi, A., Fawzi, O., Frossard, P.: Universal adversarial perturbations. CVPR (2017)
17. Pei, K., Cao, Y., Yang, J., Jana, S.: Deepxplore: Automated whitebox testing of deep learning systems. In: SOSP (2017)
18. Wang, Z., Bovik, A.C.: A universal image quality index. IEEE Signal Processing Letters **9**(3), 81–84 (March 2002). https://doi.org/10.1109/97.995823